Satellite Shade Percentage Requirements

Version 1.1

JBOP Solutions

Ben Johnson

Rachel Pang

Steffen Olson

Taylor Born

Last revised: 12/8/2013

**Table of Contents:**

1. Introduction 2
2. Problem and Solution Statement 3-4
3. Functional Requirements 5-7
	1. User Functionality 5-6
	2. Technical Tools 6-7
4. Environmental Requirements 8
5. Non-Functional Requirements 9
6. Potential Risks 10
7. Project Plan 11-12
8. **Introduction**

For this capstone project, the issue surrounds the current process of calculating shade percentage on a building prior to installing solar modules. Through talking with Seth Holland, co-founder of Rooftop Solar, our project will encompass the calculation of percentage of shade at a specific location over the course of a year where solar modules may be placed, without the need for a worker to come on-site and do it manually. The current system is inefficient and requires more manpower and company time, both a company expense as well as a safety concern for workers. Therefore a new system needs to be developed that can accurately calculate the percentage shade without the need of a worker going on-site to calculate it for each potential contract. This is where our project comes in, and is planned to use an implementation using HTTP, CSS, MySQL, Javascript, and the Google Maps API to achieve this. We are planning on implementing a user account system, as well as a project database to keep the information organized and available for use whenever needed. This will be used for managing customer cases and projects for a specialized location, as well as saving them on the user’s account. Once a Google Maps image is loaded, the user can provide on-site obstruction details, such as buildings or trees, and use the information specified on the satellite image with a developed algorithm to process the calculation. The key risk is for the algorithm to be accurate, to be deemed fit for use in making assessments on whether or not it is in a customer’s interest to invest in the installation of solar modules. However, if our solution can be accurate enough to compare with the results of the current process, then it can help not only Rooftop Solar, but other solar companies as well.

1. **Problem and Solution Statement**

For this project, our sponsor is Seth Holland at Rooftop Solar. His company focuses on the installation of and sunlight estimation for solar modules on a specified area. This is an important job, now and well into the future, so to preserve the environment by being more green and becoming less reliant on electrical companies for power. Other than the business aspect, including the buying, selling, and installation of solar modules, they also currently send people on-site to survey the area using handheld Solar Eye machines. These machines take position where a solar panel could possibly be placed, and check surrounding obstructions. Once the Solar Eye has all of the designated module points inputted, then it can calculate the percentage of sunlight that will hit the modules over the course of a year. Although this is accurate in estimating the percentage of sunlight, it requires a lot of time to send people on-site to use the Solar Eye. According to Seth, using on site manpower for this task is inefficient. In addition, the percentage of workmen's compensation that the workers sent onto the roofs to use the Solar Eye devices have, is greatly increased, leading for more company expenses in addition to the issue with worker safety. Therefore, he would like us to create a web-based application that will pull images from Google Maps and will find the percentage of shade through a “looking down” approach rather than a “looking up” one. For example; by using this system, any customer calling for an estimate to how effective solar modules would be on their house, would be able to get a fairly quick response as to how much sunlight will hit that property. Rooftop Solar would be able to arrive to an estimate via the web app, rather than spending a few hours sending someone on-site to survey the obstructions to get the same result.

Our system will use a variety of HTML, CSS, MySQL, and Javascript, as well as the Google Maps API. It will support a user system that will allow both: new account creation, and logging in and out of existing accounts; using MySQL for security. Individual accounts will have their own database of projects, where new and existing projects can be created. These projects will contain a geographical location, address, and an imported image from Google Maps. Once the image has been imported, the user will be able to mark: each solar module location, each obstruction location, and each tree with relative height to the solar panel. Once completing all marks, the system will use an algorithm similar to that of the Solar Eye in order to calculate the percentage of shade, which can then be used to decide whether the customer would benefit from solar modules, as well with quicker negotiation to estimate a price point.



Ultimately the system, if extensive enough, would be able to get a percentage of shade within 3% from that of going on-site, and will give a better basis on what the customer should expect. This process will also be more efficient this way since it doesn’t require on-site inspection. So companies can get offers quicker, and customers can make a decision quicker. Furthermore, if expanded with other regions besides Flagstaff, then it could be used to help other solar companies across the US and potentially the world.

1. **Functional Requirements**
	1. User Functionality
		1. User Accounts
			1. Creation - Users should be able to create new accounts to encompass projects(see 3.2.1.1) with certain information
				1. Username - User given account name
				2. Password
				3. Name - User’s name and optional company name
				4. User Level

Administrator account - Rooftop Solar employees have access to all features provided

Company account - Access to projects and solar calculation, but limited on some features (to be decided)

* + - 1. Log in - Users can log into account using username and password
			2. Log out - Users can click a button to logout of the system if currently logged in.
		1. Project Loading
			1. Project Database loaded that encompasses that user’s current projects and the ability to select one of them, or creation of a new project
				1. Project listing will be unique for each user
				2. Secure database connection, so each user will have access to only their databases.
		2. Project Creation
			1. Name of project - encompasses a user input for reference in project list at a future time.
			2. Region selection
				1. Multiple databases that hold information on the selected region
				2. Error checking that ensures the address is within the specified region to ensure correct calculation
			3. Address input - inputs address in that specific region to import from Google Maps for use (see 3.2.2.1)
		3. Google Maps Image Modification
			1. Zoom - image location can be zoomed in or out to specify the desired area for the user.
			2. Movement - image can be moved to center the image or move it to a more ideal position in the window for analysis
		4. Shade Percentage Calculation
			1. Image is imported and initially marked with current obstructions seen in the imported image (see 1.2.2.1) for user to see
			2. If available, a street view image should be displayed to aid the user with height estimations.
			3. Label Location of Solar Modules with Estimated Height through the use of a visual tool made in javascript (see 1.2.2.2).
			4. Label trees
				1. Trees can be marked on the image using a visual tool made in javascript (see 1.2.2.3).
				2. Relative height - The trees’ relative height to the module will also be inputted through user
			5. Label obstructions
				1. Other obstructions can be marked on the image using a visual tool made in javascript (see 1.2.2.4).
				2. Relative height - The obstructions’ relative height to the module will also be inputted through user
			6. Calculate - The user can click this button to input the heights and locations of obstructions and panels, into an algorithm that will output the percentage shade in a text box below.
	1. Technical Tools
		1. MySQL Integration
			1. Account Creation
				1. Storage of username, name, and optional name of company
				2. Password protected, and secure so it cannot be cracked into.
		2. Google Maps Integration
			1. Importation of an address, and grab image from that location, using average format (123 N. Main Ave. Cityville, ST 00000)
			2. Ability to confirm the image and copy it into a database
				1. Image itself is stored from Google Maps
				2. The scale of the image for use in future calculations.
			3. Ability to view a street view, if available
		3. Shade Percentage Calculation Tool (SPeC)
			1. Initial Markup of Image Imported
				1. Tool will automatically analyze image looking for common elements, such as trees and other buildings.
			2. Solar Module Marker
				1. User will specify location of solar modules, as well as their height.
			3. Tree Marker
				1. User can specify tree locations, as well as their relative height to the solar modules.
			4. Obstruction Marker
				1. User can specify locations of other obstructions in image (other buildings, large objects, etc.) along with their relative height to the solar modules.
			5. Calculation of Percentage (TO BE EXPLAINED IN FURTHER DETAIL ONCE ALGORITHM IS STARTED)
				1. Algorithm will take the heights of the solar modules, as well as the relative heights of the trees and obstructions to calculate the percentage shade for that location over the course of a year.
				2. Algorithm should calculate percentage up to 3% accuracy to SolarEye Devices.
1. **Environmental Requirements**
	1. The system must be web based.
		1. Server space will be provided as the project nears completion. During implementation we will need a temporary server. A simple LAMP server should suffice.
	2. The system must include map integration.
		1. The obvious choice for a map API is Google Maps, as it is freely available, at least until the product is actually ready to be sold, at which point an enterprise license will be necessary.
	3. Languages
		1. The Google Maps API is officially available for JavaScript, and as JavaScript is available on such a wide variety of platforms, this will be our language of choice.
		2. As this is a web-based product, some HTML and CSS will also be necessary.
		3. Some database functionality is needed, so MySQL will be our technology of choice. This requirement may change as we learn more about the server space that will be provided for us.
2. **Non-Functional Requirements**
	1. The algorithm for predicting the shade on solar modules shall produce accurate results within 3%, given the location and obstruction parameters.
		1. This margin of error is important to minimize because the calculation needs to be a suitable replacement of on-site inspections.
		2. The company wants to make accurate assessments for customers in determining whether they will have solar modules installed.
	2. The inputting interface shall have a high usability so that it is easily comprehended by all users.
		1. Inputting details about on-site obstructions such as structure boundaries and trees (with their relative heights) is a critical step in reaching a calculated result, so the process needs to be uncomplicated.
		2. The algorithm used to calculate the result will take into consideration the smallest differences so the meaning intended behind the user’s input needs to be unambiguous.
3. **Potential Risks**
	1. The potential risks for this project all revolve around the calculations made from the relative heights put into the algorithm. At the current moment, the system itself will be using a database to store calculations using the same data input, so the success of this project revolves primarily on the accuracy of the algorithm generated. A calculated result that is more than 3% off from what would be expected, is a problem since it is likely to be a difference that will affect whether or not someone should or shouldn’t invest in solar modules; so ensuring the algorithm is accurate is a must. While on-site analysis is still an option given murky results from the algorithm, there are still the risks from the existing system of sending someone on-site including liability risk and the aforementioned efficiency.
	2. The top-down approach of our solution relies on satellite images. If these images were not accurate, their usefulness would be diminished. Google Maps’ satellite images may be several years old for some regions. Obstructions like trees and even the landscape may have been drastically altered since the time the image was captured. Also the images may not be available at a close enough zoom for some regions. Confirmation from the user or resorting to an on-site inspection may need to be considered.
	3. Since we are using a system that will likely be marketed later on, the user system and database will need to be secure in order to ensure people already using the system don’t get locked out of their account, as well as unauthorized individuals not being able to get into the system. If the databases are not secure enough, it could lead to problems with people’s accounts leading to a less reliable and useful system, as well as putting the company’s reputation at stake.
4. **Project Plan**

Once the requirements and designs are finalized, the project plan will be as follows.

1. Develop a prototype for the basis on how the project will work
	1. Basic web pages that allow for account creation, as well as logging in and out.
	2. Ability to import an image from Google Maps
	3. Scale display on website with basic calculations from image using API and Javascript to show integration of HTML, CSS, Google Maps API and Javascript
2. Extended functionality added to user system
	1. Name/Company name, Password
	2. Security to ensure passwords are protected
	3. Implementation of a database to store users with desired information
3. Project Functionality/Database
	1. Ability to create and load projects, unique for each user.
	2. Encompasses name of project, geographical location and address.
	3. Ability to modify image grabbed from Google Maps for specified size to store.
4. User modification of Image
	1. Ability for user to mark modules, trees and obstructions on image using Javascript
	2. Each will mark relative height to module
	3. Distinct tools and colors in order to make it user friendly
5. Algorithm for Calculating the Percentage Shade implemented
	1. Uses relative heights to modules in an algorithm developed in the meantime to calculate percentage shade. (Within 2-3% of Sun Eye tool)
	2. Inputs stored in a database with the calculation to avoid multiple recalculations of same input
	3. Allows for user to make more modifications to image and calculate a new percentage based on the changes.
6. Beyond the Basic Implementation
	1. Support for as many modules and obstructions needed ( >1-2 obstructions/trees)
	2. Automatic detection of trees and obstructions as image is confirmed.
	3. Support for areas outside Flagstaff

These will be done in the current order, while working on the algorithm with Seth at the same time. Depending on the complexity of each individual milestone, it may be worked on as a group for one specific feature, or split between the members of the group.